
Eclipse 3.01

Eine Einführung

1	Einführung in Eclipse	3
1.1	Download und erster Start	3
1.2	Eclipse konfigurieren.....	5
1.2.1	Begrifflichkeiten	5
2	Projekte erstellen	5
2.1	Klasse erstellen	6
2.2	Aufbau der Workbench.....	7
2.2.1	Package Explorer	8
2.2.2	Java-Editor.....	9
2.2.3	Outline-Fenster.....	9
2.2.4	Navigator-Fenster	9
2.2.5	Problems-Fenster.....	10
2.3	Sonstiges	10
3	Anwendungen übersetzen und ausführen	10
3.1	Anwendungen übersetzen	10
3.2	Anwendungen ausführen	11
3.2.1	Anwendung schnell ausführen.....	11
3.2.2	Anwendung über eine Konfiguration starten	11
4	Eine Bibliothek erstellen und verwenden	13
4.1	Projekt erstellen	13
4.2	Klasse erstellen.....	14
4.3	JAR-File erstellen.....	17
4.4	Verwenden von JAR-Dateien	18
5	Tipps	19
6	Refactoring.....	21
6.1	Bezeichner umbenennen.....	21
6.2	Methoden verschieben	22
6.3	Pull up - Member in Basisklassen verschieben	22
6.4	Schnittstellen extrahieren	23

1 Einführung in Eclipse

1.1 Download und erster Start

Die Webseite von Eclipse finden Sie unter <http://www.eclipse.org/>. Die aktuelle Version von Eclipse ist die 3.02 (23.05.2005). Eclipse laden Sie über den Link Eclipse SDK 3.0.2 oder direkt unter

```
http://www.eclipse.org/downloads/download.php?file=/eclipse/downloads/drops/R-3.0.2-200503110845/eclipse-SDK-3.0.2-win32.zip
```

Die Installation erfolgt durch das einfache Entpacken der ZIP-Datei in einen Ordner Ihrer Wahl. In diesem Ordner wird ein Verzeichnis `..eclipse` erzeugt. Darin finden Sie die Datei `eclipse.exe`, welche Eclipse unter Windows startet.

Nach den erstmaligen Start von Eclipse werden Sie gefragt, wo Ihr Arbeitsplatz eingerichtet werden soll. Standardmäßig wird das Unterverzeichnis `..workspace` Ihrer Eclipse-Installation vorgeschlagen. Damit Sie nicht jedes mal beim Start danach gefragt werden, markieren Sie die Option `USE THIS AS THE DEFAULT AND DO NOT ASK AGAIN`.

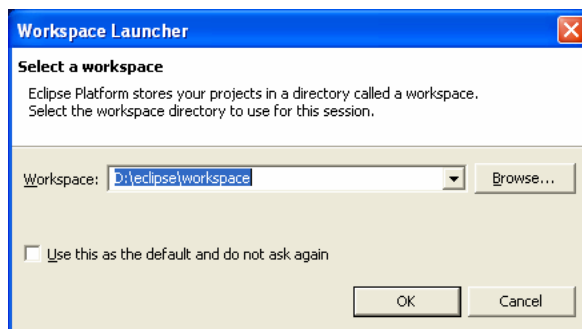


Abb. 1.1 Workspace auswählen

Nach der Bestätigung mit OK wird Eclipse mit der Welcome-Seite geöffnet.

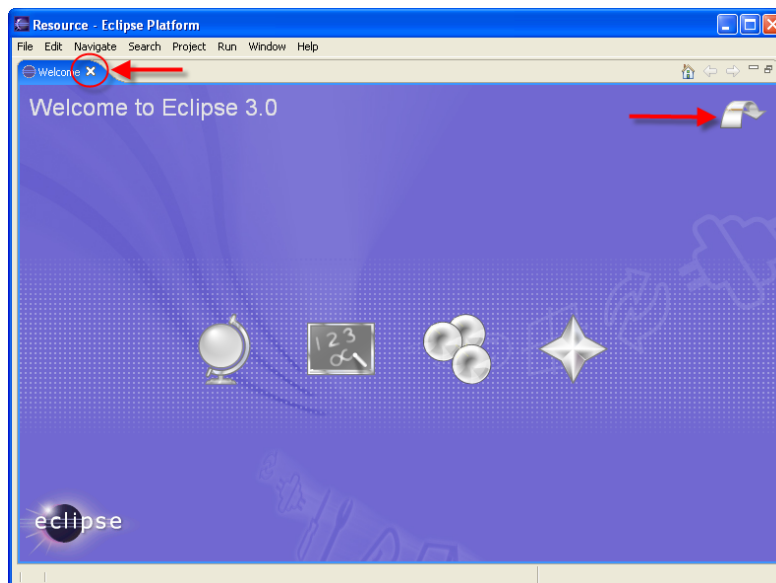
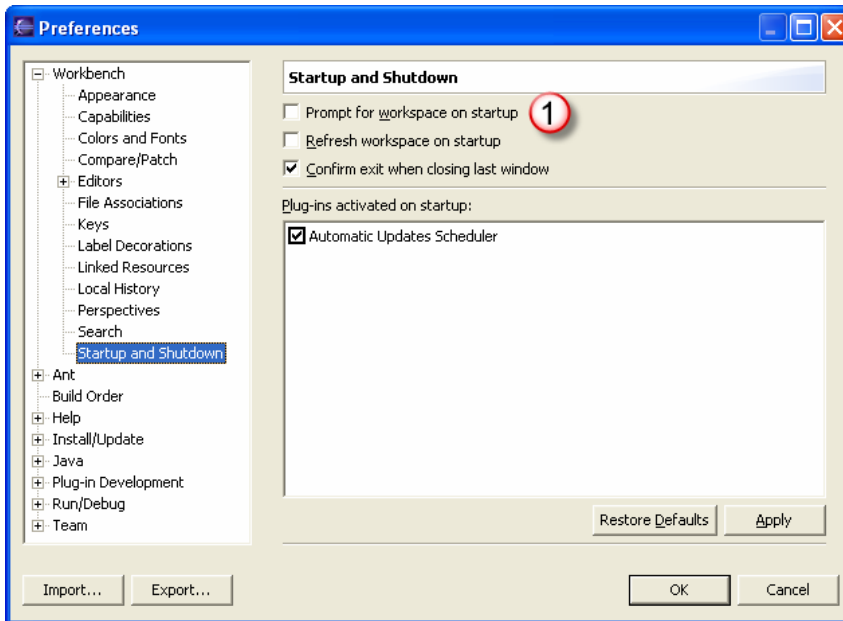


Abb. 1.2 Eclipse-Willkommenseite



Die Anzeige des Dialogfeldes zur Auswahl des Workspaces stellen Sie unter dem Menüpunkt WINDOW - PREFERENCES, Kategorie WORKBENCH - STARTUP AND SHUTDOWN ein.



Die Welcome-Seite enthält in der Mitte 4 Symbole, über die Sie erste Kontakte mit Eclipse knüpfen können. Die Willkommenseite schließen Sie entweder über das Kreuz auf dem Registerreiter links oder den Pfeil oben rechts.



Die Willkommenseite können Sie jederzeit über den Menüpunkt HELP - WELCOME anzeigen.

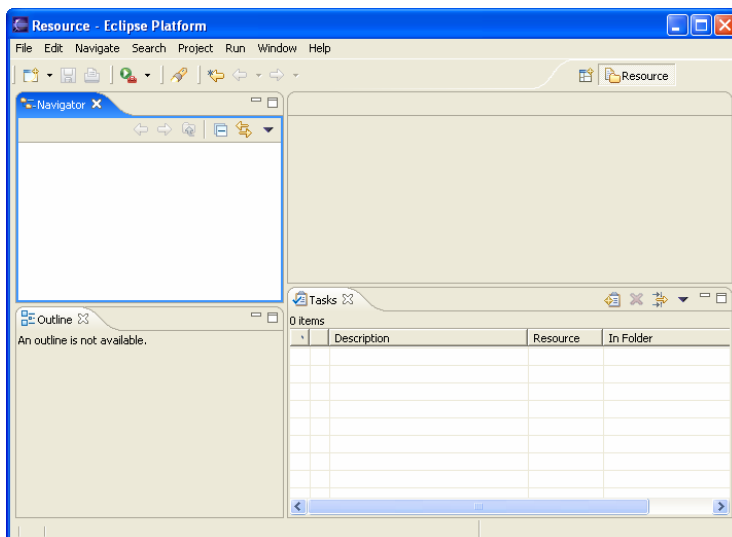


Abb. 1.3 Eclipse-IDE ohne ein geöffnetes Projekt

Die Anzeige der Workbench ist nach dem ersten Start sehr aufgeräumt. Um sie mit Leben zu füllen, müssen Sie ein Projekt erstellen. Vorher werden aber erst ein paar Einstellungen vorgenommen.



Den aktuellen Workspace ändern Sie über den Menüpunkt FILE - SWITCH WORKSPACE.

1.2 Eclipse konfigurieren

Rufen Sie den Menüpunkt WINDOW - PREFERENCES auf.

Kategorie / Subkategorie	Einstellung - Beschreibung
WORKBENCH	BUILD AUTOMATICALLY - Beim Speichern werden die Projekte automatisch aktualisiert / übersetzt.
	ALWAYS RUN IN BACKGROUND - Lang anhaltende Operationen werden im Hintergrund ausgeführt.
JAVA - CODE STYLE - CODE FORMATTER	Klicken Sie auf SHOW um einen erweiterten Dialog anzuzeigen. Interessant ist die Angabe der Tabulatorbreite auf dem Register INDENTATION und die Angabe der Ausrichtung der Klammern unter BRACES. Nachdem Sie Änderungen durchgeführt haben werden Sie aufgefordert einen neuen Namen für das Profil anzugeben.
WORKBENCH - JAVA - EDITOR	DISPLAY TAB WIDTH - Tabulatorbreite
	SHOW LINE NUMBERS - Zeilennummern anzeigen
WORKBENCH - JAVA - INSTALLED JRES	Hier können Sie ein weiteres JRE hinzufügen.

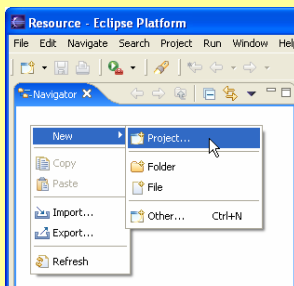
Einstellungen können über die Schaltflächen EXPORT und IMPORT für andere Mitarbeiter bereitgestellt werden.

1.2.1 Begrifflichkeiten

Workbench	Die Eclipse-IDE wird als Workbench bezeichnet.
Workspaces	Ein Workspace ist ein Verzeichnis, in dem Ihre Projekte verwaltet werden.
Perspektiven	Eine Perspektive ist eine Sammlung von Einstellungen und Fensteranordnungen der Workbench.
Projekte	Ein Projekt kann separat konfiguriert werden und ist Bestandteil des aktuellen Workspaces.

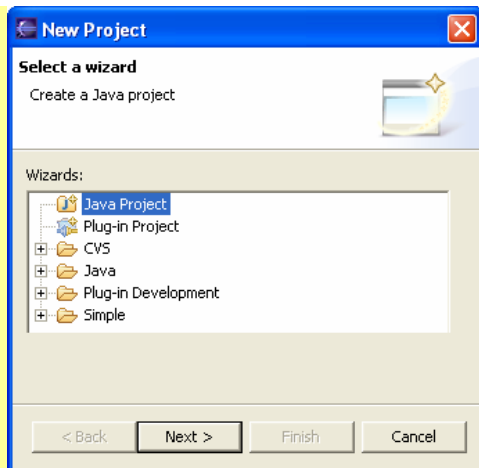
2 Projekte erstellen

- ✘ Klicken Sie im Kontextmenü des Navigatorfensters auf den Menüpunkt NEW - PROJECT.
- ✘ Alternativ rufen Sie den Menüpunkt FILE - NEW - PROJECT des Hauptmenüs auf.

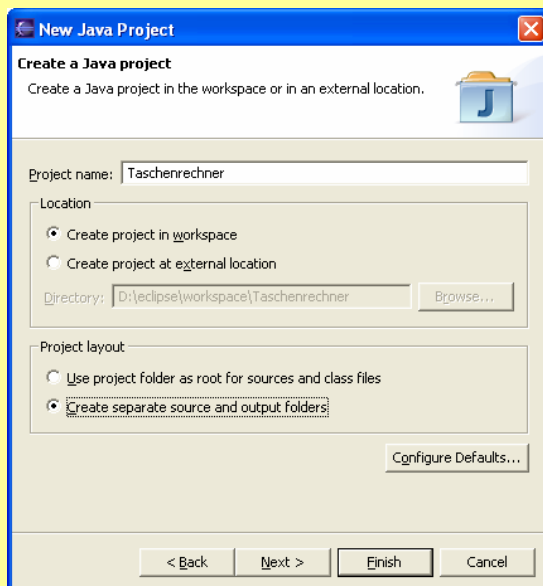


- ✘ Markieren Sie im Fenster New Project den Eintrag JAVA PROJECT und klicken Sie auf NEXT.

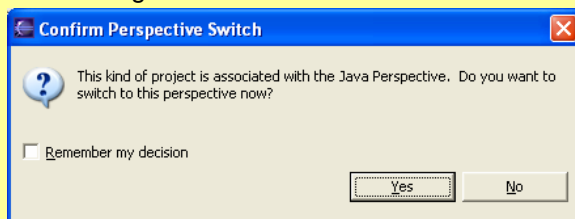
Projekte erstellen



- ✘ Vergeben Sie einen Projektnamen (z.B. Taschenrechner), markieren Sie die Option CREATE PROJECT IN WORKSPACE (um das Projekt im aktuellen Workspace zu erstellen) und markieren Sie außerdem die Option CREATE SEPARATE SOURCE AND OUTPUT FOLDERS (um getrennte Ordner für den Java-Code und die Class-Dateien zu erhalten).



- ✘ Klicken Sie auf Finish.
- ✘ Es wird ein Hinweisfenster angezeigt, dass dieser Projekttyp mit der Java-Perspektive verknüpft ist. Bestätigen Sie mit Yes.

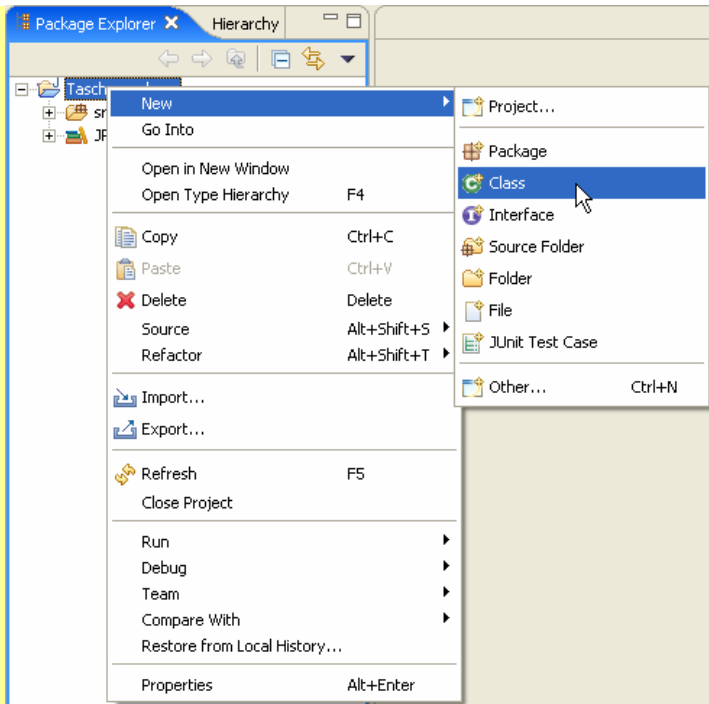


- ✘ Damit ist die Projekterstellung abgeschlossen. Bevor die weiteren Teile der Workbench besprochen werden, soll eine Klasse erzeugt werden.

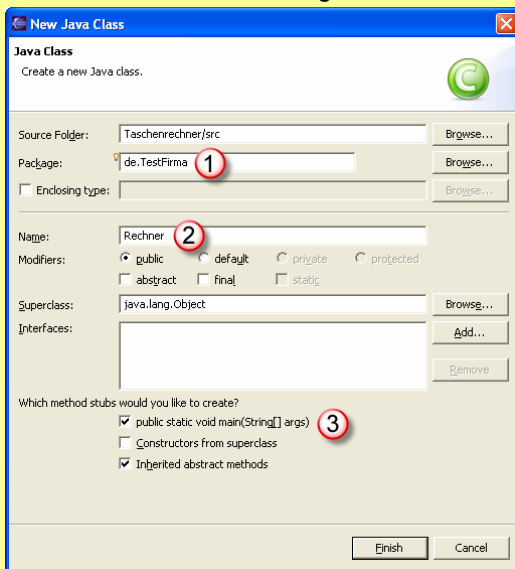
2.1 Klasse erstellen

- ✘ Rufen Sie im Package Explorer den Kontextmenüpunkt NEW - CLASS auf.
- ✘ Alternativ verwenden Sie den Menüpunkt FILE - NEW - CLASS.

Projekte erstellen



- ✘ Im Dialogfenster NEW JAVA CLASS geben Sie bei (1) optional einen Packagenamen an und unter (2) den Klassennamen. Wenn die Klasse eine `main()`-Methode besitzen soll, machen Sie bei (3) noch einen Haken. Bestätigen Sie mit FINISH.

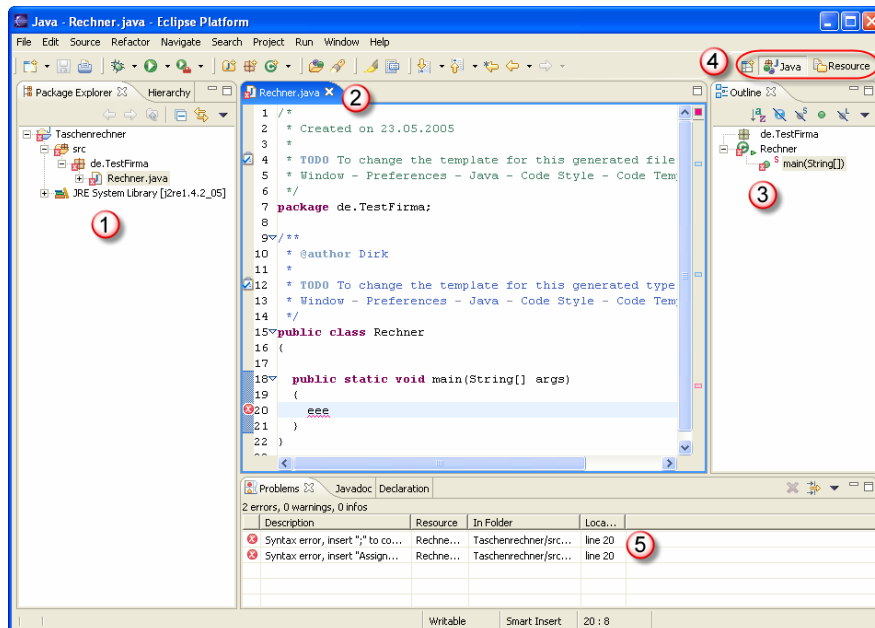


- ✘ Die neue Klasse wird im Java-Editor geöffnet. Jetzt ist ein guter Zeitpunkt, die einzelnen Fenster der Workbench näher zu betrachten.

2.2 Aufbau der Workbench

Ist ein Projekt geöffnet bzw. sind Projekte im Workspace verfügbar, kommt etwas mehr Leben in die Workbench. Die folgende Abbildung zeigt die Fenster, die in der Java-Perspektive sichtbar sind. Im aktiven Fenster ist das Register blau gefärbt.

Projekte erstellen



Info

Unter (4) können Sie momentan zwischen der Java- und der Resource-Perspektive umschalten. Weitere Perspektiven können Sie unter dem Menüpunkt WINDOW - OPEN PERSPECTIVE oder über das linke Symbol unter (4) öffnen.

2.2.1 Package Explorer

Der Package Explorer zeigt alle Projekte und Dateien des aktuellen Workspaces an. Neue Klassen werden im aktuellen Projekt im Verzeichnis *src* angelegt, da beim Erstellen des Projekts die Option CREATE SEPARATE SOURCE AND OUTPUT FOLDERS markiert wurde. Dadurch ist jetzt die folgende Verzeichnisstruktur entstanden:

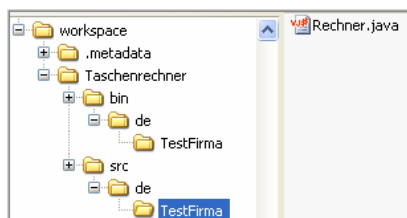


Abb. 2.4

- ✘ Im Workspace-Ordner wurde ein neuer Ordner erstellt, der den Projektnamen trägt.
- ✘ Darunter wurden zwei Verzeichnisse *src* und *bin* erstellt, welche die Source-Dateien und die übersetzten Class-Dateien enthalten. Damit haben Sie eine gute Trennung zwischen den Dateien, die evt. in einer Versionsverwaltung gesichert werden müssen, und den Dateien, die Ihre Anwendung ausmachen.
- ✘ Da im Quelltext momentan ein Fehler vorliegt, erhalten die Symbole im Package Explorer ein rotes Symbol links unten.

Info

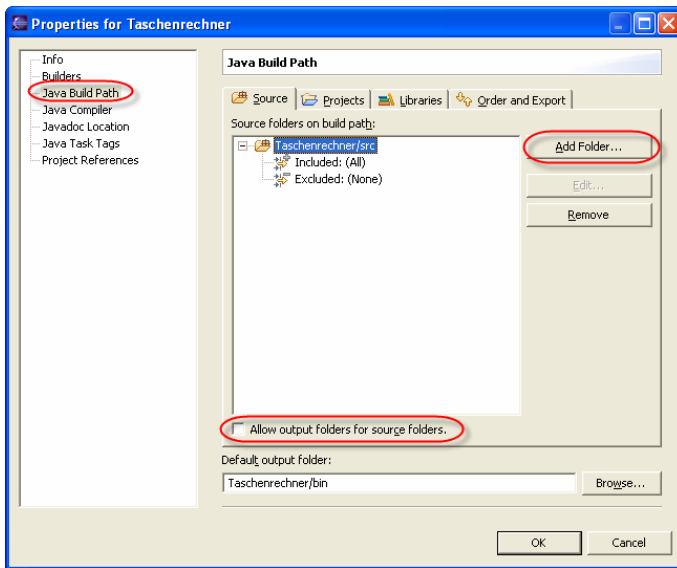
Der Package Explorer zeigt nur die für das Projekt notwendigen Dateien an. Erzeugte Class-Dateien oder das *bin*-Verzeichnis werden nicht angezeigt.

Info

Unter dem Menüpunkt PROJECT - PROPERTIES können Sie die Ordnerstruktur für Source- und Ausgabeordner jederzeit ändern. Markieren Sie die Option ALLOW OUTPUT ..., um keine getrennten Ordner zu verwenden. Haben Sie dagegen nur einen Ordner angelegt, können Sie

Projekte erstellen

über ADD FOLDER einen neuen Ordner anlegen. Eclipse erzeugt dann automatisch einen *bin*-Ordner für die Ausgabe.



2.2.2 Java-Editor

Der Java Editor dient zur Eingabe des Quelltextes. Sind mehrere Dateien geöffnet, werden standardmäßig mehrere Registerkarten angezeigt.

2.2.3 Outline-Fenster

Dieses Fenster dient zur Navigation im Source-Code im aktuell geöffneten Editorfenster. Es werden die Typen (Klassen, Interfaces), Methoden usw. angezeigt. Klicken Sie auf einen Eintrag, wird er im Editor selektiert. Umgekehrt wird der Fensterinhalt mit der aktuellen Position im Editor synchronisiert.

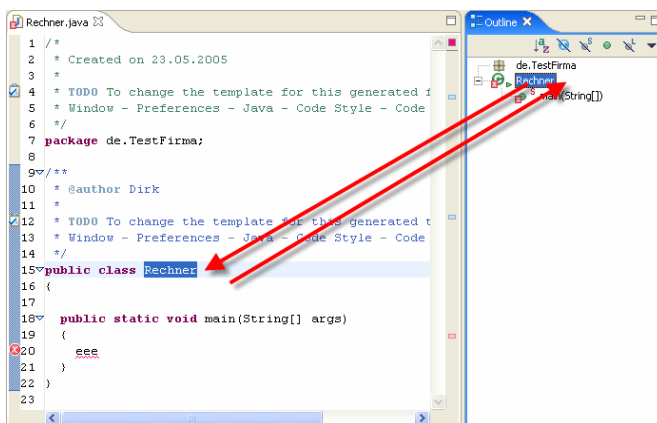


Abb. 2.5

2.2.4 Navigator-Fenster

In der Java-Perspektive wird das Navigatorfenster nicht angezeigt. Dazu wechseln Sie entweder in die Perspektive Resource oder Sie rufen den Menüpunkt WINDOW - SHOW VIEW - NAVIGATOR auf. Das Fenster zeigt alle Dateien des Workspaces an. Über das Kontextmenü können Sie diese auch bearbeiten, z.B. umbenennen.

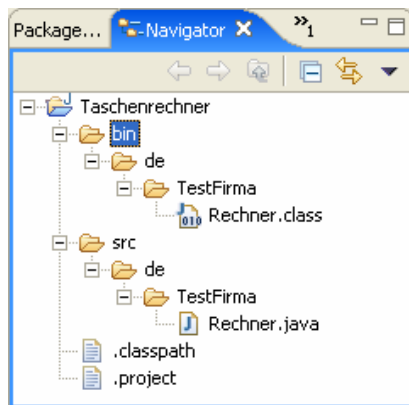


Abb. 2.6

2.2.5 Problems-Fenster

Wurden Fehler festgestellt, werden diese in diesem Fenster angezeigt. Durch einen Klick auf die Meldung gelangen Sie direkt zum fehlerhaften Code.

2.3 Sonstiges

Projekte können zwar geschlossen werden, sie werden im Package Explorer aber immer im zugeklappten Zustand angezeigt. Allerdings beanspruchen geschlossene Projekte weniger Speicher.

3 Anwendungen übersetzen und ausführen

Fügen Sie in die Methode `main()` den Code `System.out.println("Hallo");` ein.

```
public class Rechner
{
    public static void main(String[] args)
    {
        System.out.println("Hallo");
    }
}
```



Obwohl der Code jetzt fehlerfrei ist, werden immer noch die Fehlermeldungen angezeigt. Erst wenn Sie die Datei speichern wird automatisch der Code im Hintergrund übersetzt und die Anzeige aktualisiert. Dadurch liegt auch sofort ein fertig übersetztes Programm vor.

3.1 Anwendungen übersetzen

Haben Sie unter WINDOW - PREFERENCES, Kategorie WORKBENCH die Option BUILD AUTOMATICALLY aktiviert, wird bei jedem Speichern das Projekt aktualisiert und übersetzt. Alternativ können Sie auch den Haken unter dem Menüpunkt PROJECT - BUILD AUTOMATICALLY setzen. Ist der Haken nicht gesetzt, stehen die Menüpunkte PROJECT - BUILD ALL und PROJECT - BUILD PROJECT zur Verfügung.

3.2 Anwendungen ausführen

Damit eine Anwendung ausgeführt werden kann, muss normalerweise eine Konfiguration dazu erstellt werden. Allerdings kann eine Anwendung auch ohne diesen "Umweg" ausgeführt werden.

3.2.1 Anwendung schnell ausführen

Klicken Sie auf einen Knoten im Package Explorer und rufen Sie den Kontextmenüpunkt RUN - JAVA APPLICATION auf. Eclipse sucht nun in allen Dateien des Knotens und seinen Unterknoten nach Class-Dateien, die eine `main()`-Methode enthalten, also ausgeführt werden können. Wenn Sie z.B. auf den Projektknoten klicken, wird dazu das Projekt aber auch das gesamte JRE durchsucht.

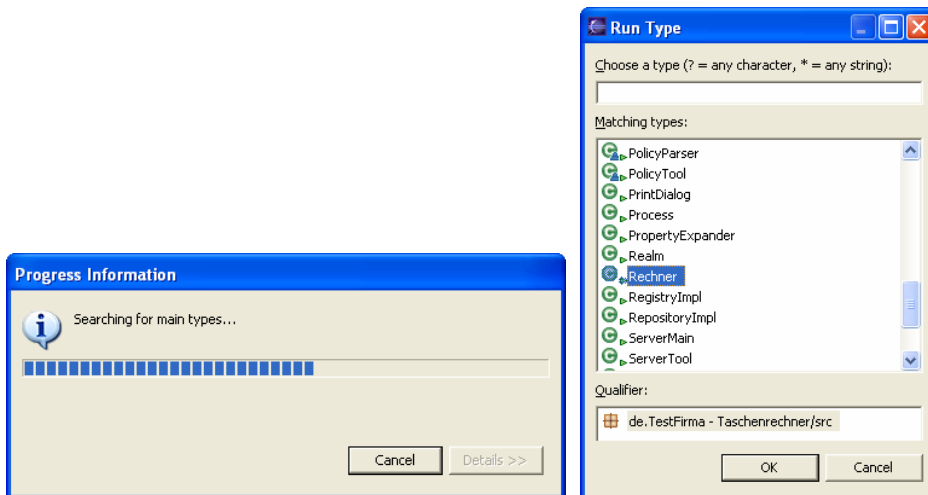

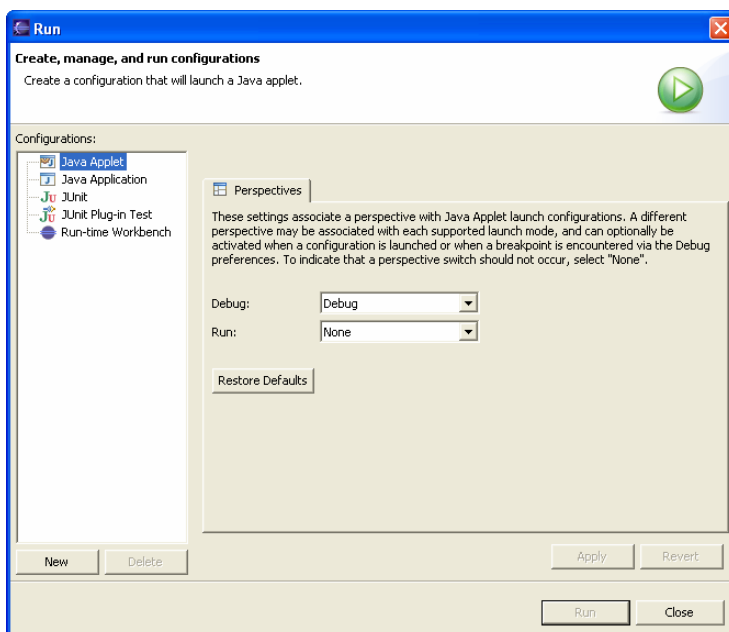


Abb. 3.7

Deshalb ist es besser gleich die entsprechende Source-Datei auszuwählen und die Anwendung darüber zu starten.

3.2.2 Anwendung über eine Konfiguration starten

Haben Sie auf den Run-Button () in der Symbolleiste geklickt oder den Menüpunkt RUN - RUN aufgerufen, wird das folgende Fenster geöffnet.



Anwendungen übersetzen und ausführen

Abb. 3.8



Haben Sie das Projekt bereits vorher einmal gestartet befindet sich bereits eine Konfiguration unter dem Eintrag JAVA APPLICATION.

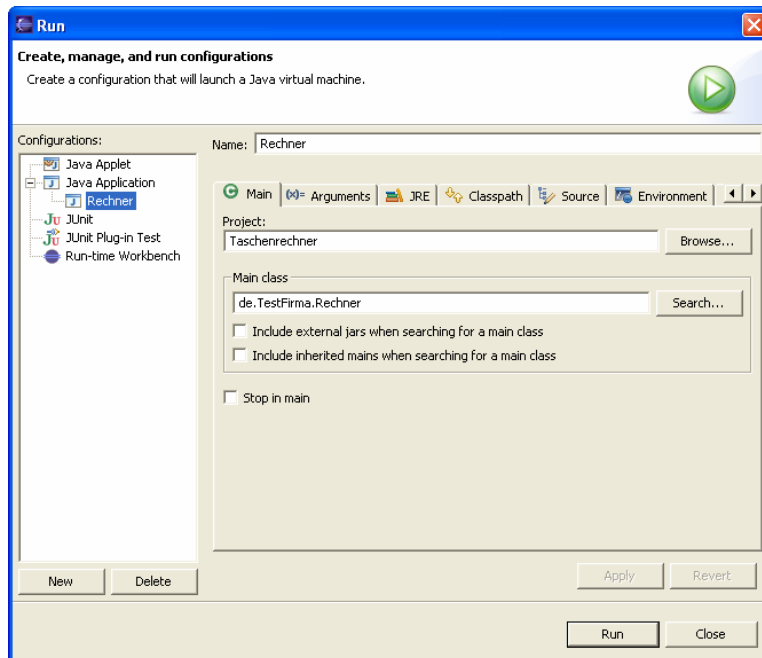
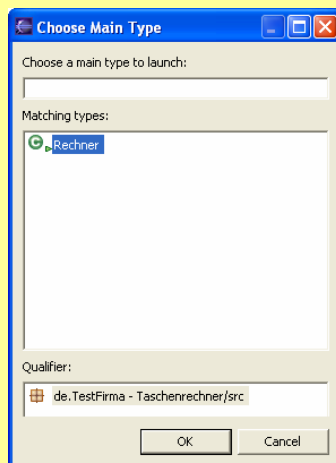


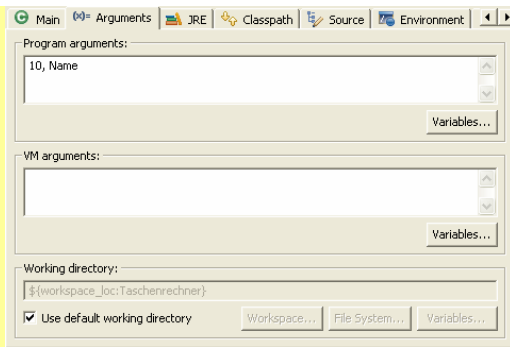
Abb. 3.9

- ✘ Klicken Sie im Dialog aus Abbildung 3.7 auf den Eintrag JAVA APPLICATION.
- ✘ Klicken Sie auf NEW.
- ✘ Vergeben Sie im Feld NAME einen Namen für die Konfiguration, z.B. Taschenrechner.
- ✘ Klicken Sie auf SEARCH, um die Hauptklasse Ihrer Anwendung auszuwählen und bestätigen Sie mit OK.



- ✘ Über die Register ARGUMENTS, CLASSPATH und SOURCE können Sie Parameter an die Anwendung übergeben, den Klassenpfad definieren und weitere Ordner mit SourceCode hinzufügen.

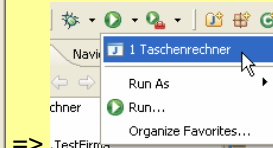
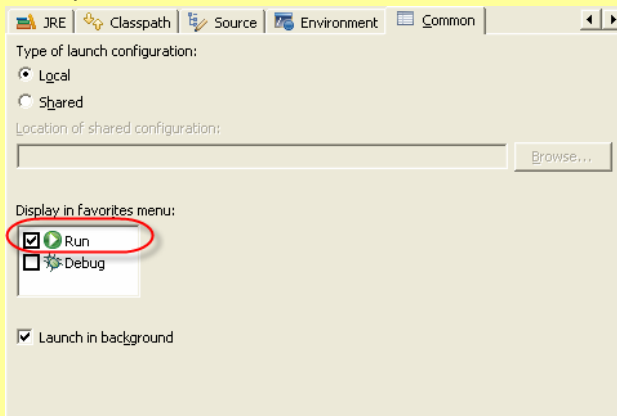
Eine Bibliothek erstellen und verwenden



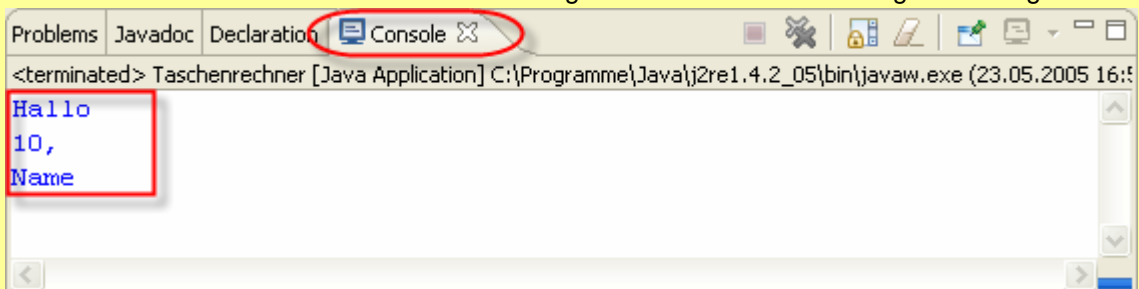
- ✘ Ändern Sie dazu den Code der Methode `main()` in

```
public static void main(String[] args)
{
    System.out.println("Hallo");
    for(int i = 0; i < args.length; i++)
        System.out.println(args[i]);
}
```

- ✘ Im letzten Register COMMON markieren Sie die Option RUN, damit die Konfiguration als Menüpunkt bei der Schaltfläche RUN erscheint.



- ✘ Starten Sie jetzt die Anwendung. Da Sie Ausgaben auf der Console durchgeführt haben, wird im unteren Bereich ein weiteres Fenster CONSOLE geöffnet. Darin wird die Ausgabe durchgeführt.



4 Eine Bibliothek erstellen und verwenden

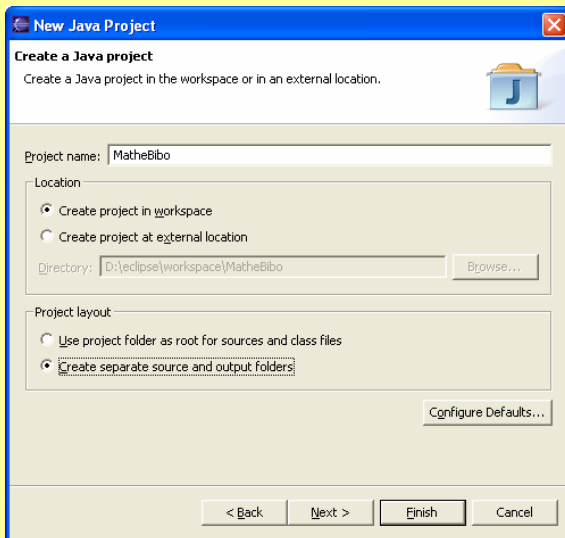
Jetzt soll ein Archiv erstellt werden, das eine Klasse `Mathe` enthält, die Methoden zum Addieren und Subtrahieren enthält. Das Archiv soll in der Anwendung `Rechner` genutzt werden.

4.1 Projekt erstellen

- ✘ Rufen Sie den Menüpunkt `FILE - NEW - PROJECT` auf.
- ✘ Wählen Sie `JAVA PROJECT` und klicken Sie auf `NEXT`.

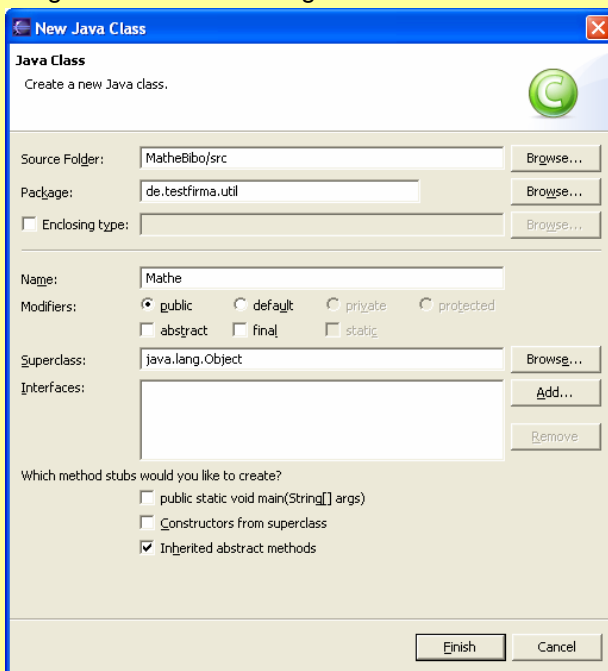
Eine Bibliothek erstellen und verwenden

- ✘ Nennen Sie das Projekt MATHEBIBO, markieren Sie CREATE SEPARATE ... und bestätigen Sie mit FINISH.



4.2 Klasse erstellen

- ✘ Rufen Sie im Kontextmenü des neuen Projekts im Package Explorer den Menüpunkt NEW - CLASS auf.
- ✘ Vergeben Sie den Packagenamen `de.testfirma.util` und benennen Sie die Klasse `Mathe`.

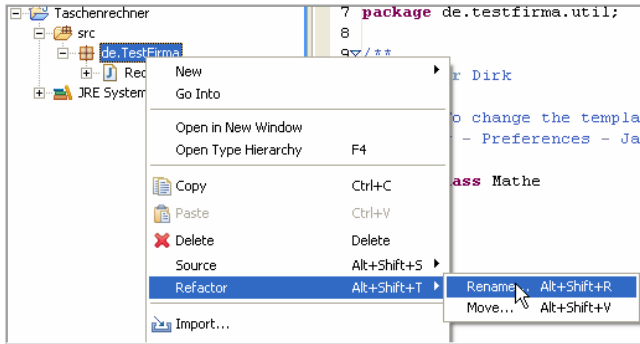


- ✘ Bestätigen Sie mit FINISH.

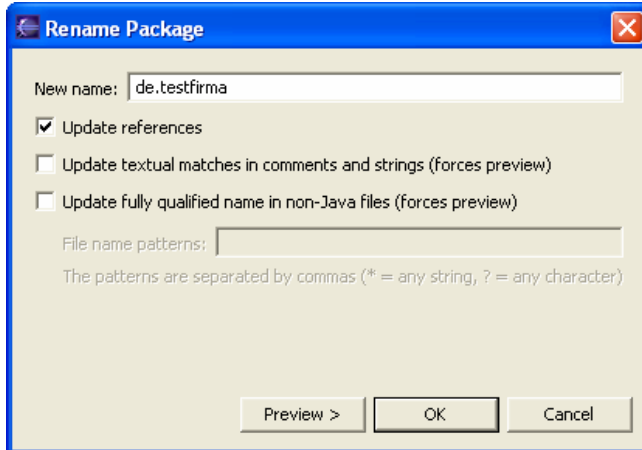


Beim Erstellen der Klasse ist Ihnen aufgefallen, dass der Packagename der Hauptanwendung nicht den allgemeinen Konventionen für Packagenamen entspricht. Diese sollten nämlich immer klein geschrieben werden. Klicken Sie auf das Packagesymbol `de.TestFirma` und rufen Sie den Menüpunkt REFACTOR - RENAME auf.

Eine Bibliothek erstellen und verwenden



Geben Sie `de.testfirma` ein und bestätigen Sie mit OK.



Ignorieren Sie die angezeigte Meldung und bestätigen Sie mit CONTINUE. Neben den Verzeichnisnamen wurde auch die `package`-Anweisung in der Datei `Rechner.java` aktualisiert.

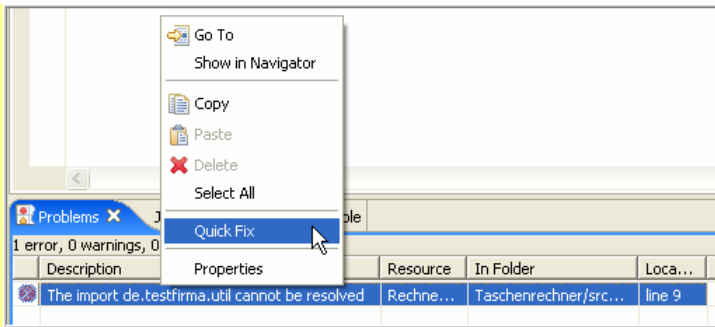
Nach diesem kleinen Ausflug geht es weiter mit der Erstellung der Bibliothek.

- ✘ Implementieren Sie die Klasse `Mathe` wie folgt.

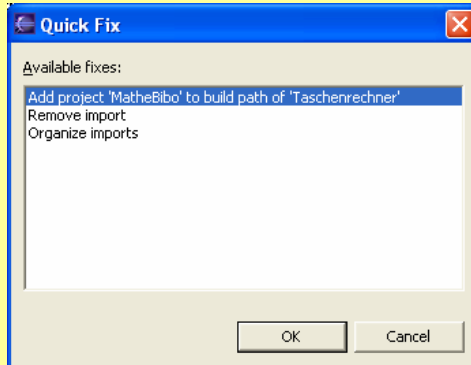
```
public class Mathe
{
    public static int Add(int zahl1, int zahl2)
    {
        return zahl1 + zahl2;
    }
    public static int Minus(int zahl1, int zahl2)
    {
        return zahl1 - zahl2;
    }
}
```

- ✘ Fügen Sie in der Klasse `Mathe` des Projekts `Taschenrechner` die folgende `import`-Anweisung ein.
`import de.testfirma.util.*;`
und speichern Sie die Datei. Im Problems-Fenster wird ein Fehler angezeigt, da sich das Projekt mit der Klasse `Mathe` nicht im Klassenpfad befindet.
- ✘ Klicken Sie mit der rechten Maustaste auf die Fehlermeldung und rufen Sie den Kontextmenüpunkt `QUICKFIX` auf.

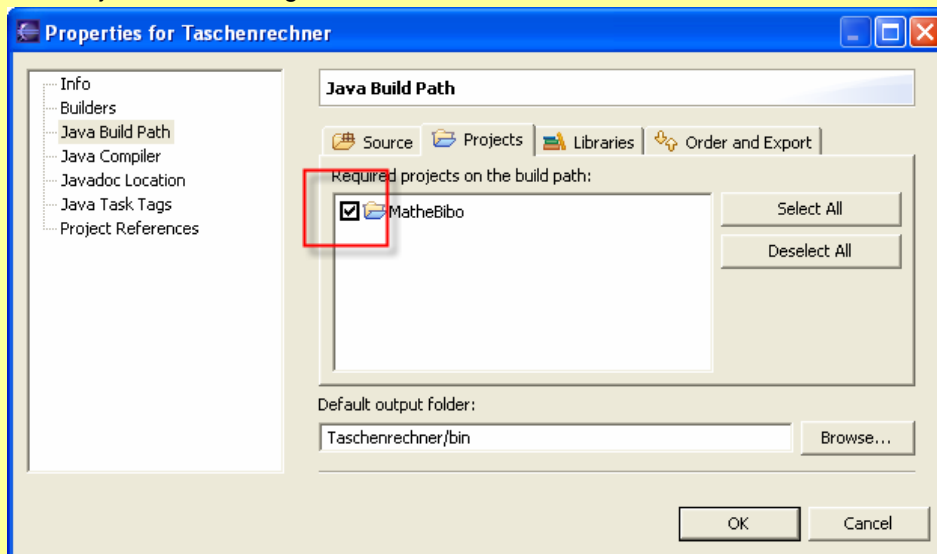
Eine Bibliothek erstellen und verwenden



- ✘ Im Dialogfenster QUICK FIX werden drei Lösungsmöglichkeiten für das Problem vorgeschlagen. Die markierte Lösungsmöglichkeit beseitigt das Problem. Bestätigen Sie mit OK.

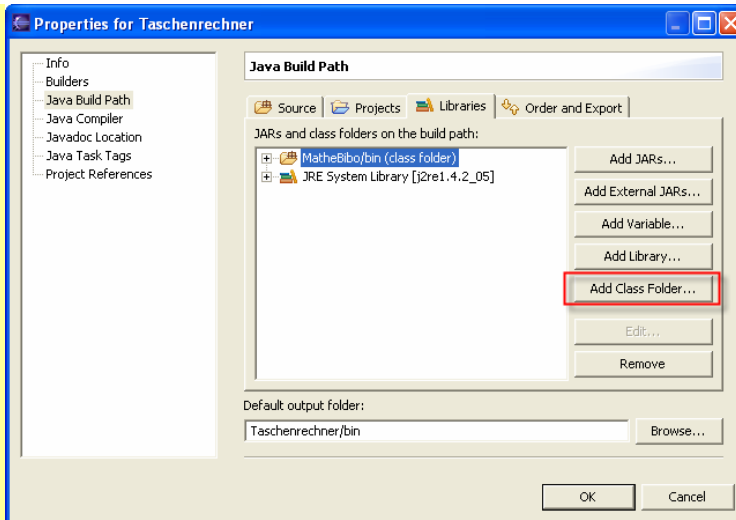


- ✘ Als Lösung wurde in den Projektoptionen (Menüpunkt PROJECT - PROPERTIES) ein Haken vor das Projekt MatheBibo gesetzt.

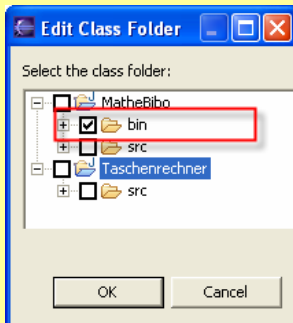


- ✘ Alternativ können Sie auch in das Register LIBRARIES wechseln und klicken dort auf ADD CLASS FOLDER.

Eine Bibliothek erstellen und verwenden



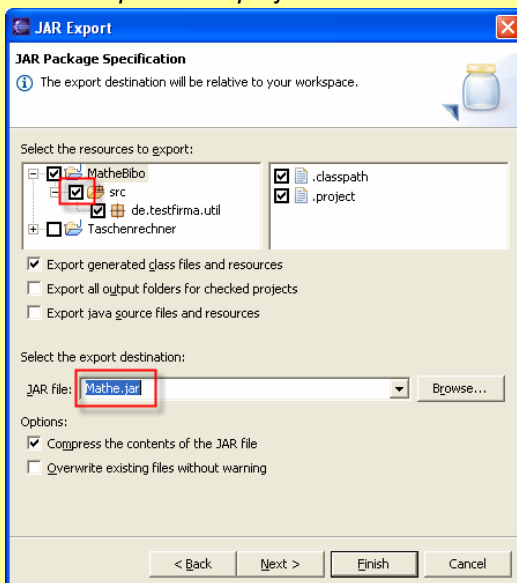
- ✘ Setzen Sie einen Haken vor das *bin*-Verzeichnis des Mathe-Projekts.



- ✘ Fügen Sie in die `main()`-Methode noch die beiden Anweisungen ein:
`System.out.println(Mathe.Add(10, 11));`
`System.out.println(Mathe.Minus(11, 34));`

4.3 JAR-File erstellen

- ✘ Rufen Sie im Package Explorer den Kontextmenüpunkt EXPORT auf.
- ✘ Wählen Sie als Exportformat JAR FILE und klicken Sie NEXT.
- ✘ Markieren Sie das `src`-Verzeichnis des Mathe-Projekts. Geben Sie in das Feld JAR FILE den Namen des neuen Archivs ein, z.B. *Mathe.jar*. Deaktivieren Sie gegebenenfalls die Markierungen vor `.classpath` und `.project`.



Eine Bibliothek erstellen und verwenden

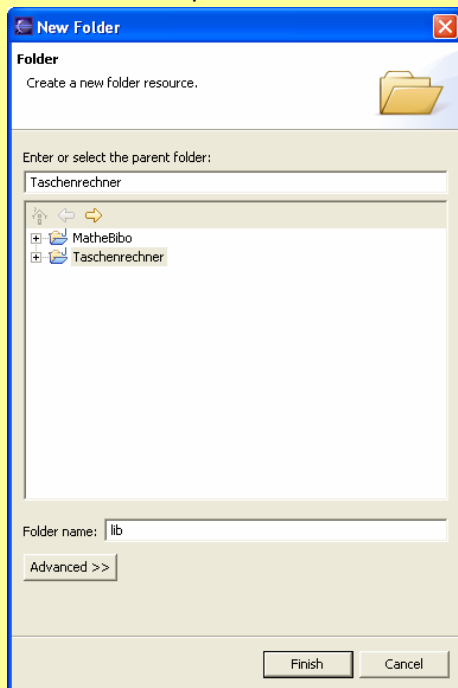
✘ Klicken Sie auf FINISH. Die JAR-Datei wird im Workspace-Verzeichnis erstellt.

 Ein umfangreicheres Buildmanagement ist mit Ant möglich.

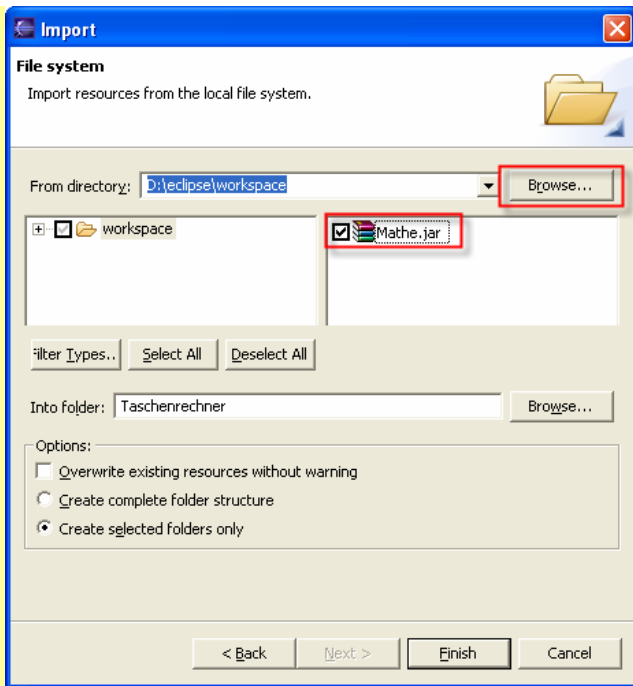
4.4 Verwenden von JAR-Dateien

Statt nun das Verzeichnis der Class-Dateien des Mathe-Projekts zu referenzieren soll das erstellte JAR-Archiv genutzt werden. Dies ist beispielsweise der Standardvorgang, wenn Sie Archive von Fremdherstellern nutzen wollen.

- ✘ Entfernen Sie aus dem Rechner-Projekt die Referenzen auf das Mathe-Projekt.
- ✘ Erstellen Sie über den Package Explorer einen neuen Ordner *lib* im Rechner-Projekt. Rufen Sie dazu den Menüpunkt NEW - FOLDER des Projekt auf.

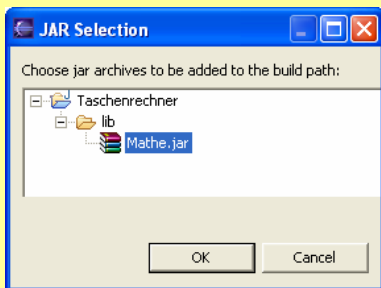


- ✘ Rufen Sie im Package Explorer den Kontextmenüpunkt IMPORT des Ordner *lib* im Rechner-Projekts auf.
- ✘ Wählen Sie den Eintrag FILE SYSTEM und klicken Sie auf NEXT.
- ✘ Wählen Sie nach dem Klick auf BROWSE den Ordner mit dem Archiv aus und markieren Sie es.



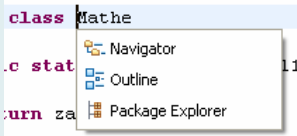
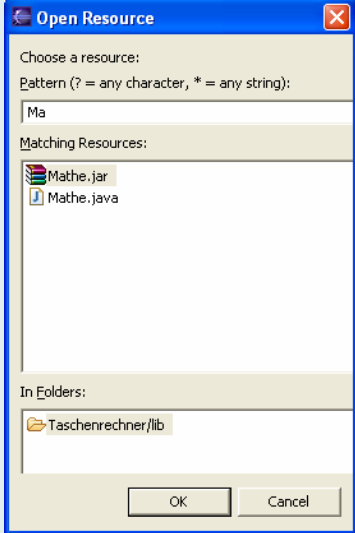

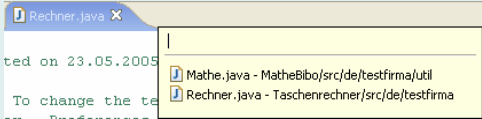
✘ Bestätigen Sie mit FINISH.

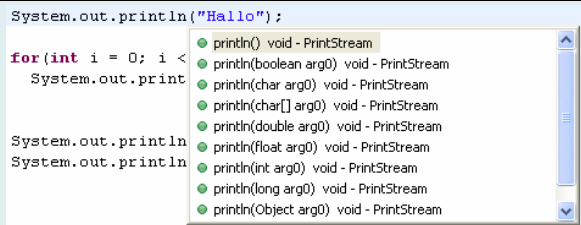
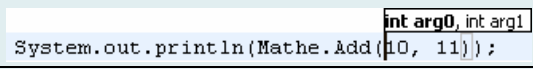
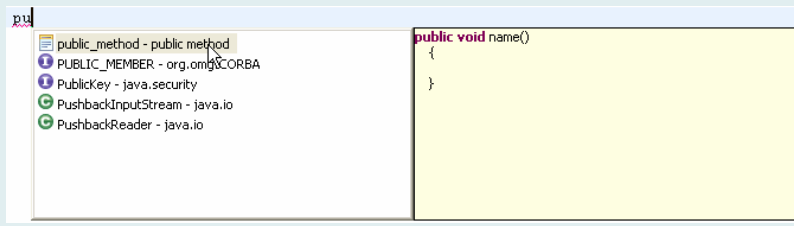
- ✘ Jetzt muss noch das Archiv dem Klassenpfad des Projekts hinzugefügt werden.
- ✘ Öffnen Sie die Projektoptionen über den Menüpunkt PROJECT - PROPERTIES, markieren Sie die Kategorie JAVA BUILD PATH und wechseln Sie in das Register LIBRARIES.
- ✘ Klicken Sie auf ADD JARS. Wählen Sie dann das importierte Archiv aus und bestätigen Sie mit OK.



5 Tipps

Navigationshistorie	Über ALT + ⇐ und ALT + ⇒ können Sie sich im Editor vorwärts und rückwärts entsprechend Ihrer eigenen Navigation bewegen.
Inkrementelle Suche	Über STRG + J starten Sie die inkrementelle Suche. In der Statuszeile wird <u>Incremental Find</u> angezeigt. Geben Sie jetzt die nacheinander die Anfangsbuchstaben des Suchbegriffes ein. Diese werden dann in der Statuszeile angezeigt und im Editor wird die erste Fundstelle markiert. Mit STRG + J bewegen Sie sich zur nächsten Fundstelle.
Synchronisation	Wenn Sie sich auf einem Begriff befinden klicken Sie UMSCHALT + ALT + W. Es wird ein Popup angezeigt, über das Sie mit dem jeweiligen View synchronisieren können.

	
Resource-Suche	<p>Um schnell eine Ressource zu finden, betätigen Sie STRG + UMSCHALT + R. Geben Sie nun die Anfangsbuchstaben der Ressource ein.</p> 
Mehrere Editorfenster	<p>Ziehen Sie das Register des Editors an eine "Kante". Dann können Sie mehrere Editorfenster nebeneinander sehen. Um es wieder anzudocken, ziehen Sie es auf den Tab eines anderen Fensters.</p>
Alles zuklappen	<p>Klicken Sie auf , um alle Knoten einer Ansicht zu schließen.</p>
Javadoc-Kommentare	<p>Geben Sie die Zeichenfolge <code>/**</code> ein und betätigen Sie Return. Es wird automatisch der Code analysiert und der Kommentar abgeschlossen, z.B. für die Methode <code>Minus(int zahl1, int zahl2)</code> wird erzeugt:</p> <pre data-bbox="448 1263 1401 1462"> /** * * @param zahl1 * @param zahl2 * @return */ </pre>
ToDo-Listen	<p>Geben Sie in einem Javadoc-Kommentar <code>ToDo</code> gefolgt von einem Text ein. Dieser Text wird im Task-Fenster angezeigt (WINDOW - SHOW VIEW - OTHER - TASKS).</p> <pre data-bbox="448 1585 1401 1653"> /** * TODO Hier ist noch was zu tun... </pre>
Editorliste	<p>Über <code>Strg + E</code> wird eine Liste mit allen geöffneten Editorfenstern angezeigt.</p> 
Syntaxhilfe im Editor	<p>STRG + LEERTASTE betätigen</p>

	 <pre>System.out.println("Hallo"); for (int i = 0; i < System.out.print System.out.println System.out.println</pre>
Inhaltshilfe	In einer Anweisung betätigen Sie STRG + LEERTASTE. Es werden die möglichen Zuweisungen angezeigt.
Parameterhilfe	STRG + UMSCHALT + LEERTASTE in der Parameterliste einer Methode betätigen  <pre>System.out.println(Mathe.Add(10, 11));</pre>
Codevorlagen	Geben Sie die Anfangsbuchstaben ein und betätigen sie STRG + LEERTASTE.  <pre>public void name() { }</pre> <p>Die Codevorlagen werden unter WINDOW - PREFERENCES, JAVA - EDITOR - TEMPLATES definiert.</p>
Interfaces implementieren	Rufen Sie in der Klasse den Kontextmenüpunkt SOURCE - OVERRIDE/IMPLEMENT METHODS auf.

6 Refactoring

Über Refactoring verändern Sie die Struktur Ihres Codes, ohne dessen Funktionalität zu ändern. Ziel ist es, den Code besser wartbar, lesbar und wieder verwendbarer zu gestalten. Typische Refactorings sind:

- ✘ Aussagekräftige Variablennamen
- ✘ Aufteilen langer Codepassagen in mehrere Methoden

6.1 Bezeichner umbenennen

- ✘ Begeben Sie sich zur Deklaration des Bezeichners und rufen Sie den Menüpunkt REFACTOR - RENAME auf.
- ✘ Alternativ rufen Sie den gleichnamigen Kontextmenüpunkt auf.
- ✘ Vergeben Sie im Dialogfeld einen neuen Namen. Entspricht der Name nicht den Konventionen für die Namensgebung unter Java, wird außerdem eine Meldung angezeigt.
- ✘ Durch die Markierung der Option UPDATE REFERENCES werden auch Referenzen auf den Bezeichner (Methode, Variable, Klasse etc.) umbenannt.

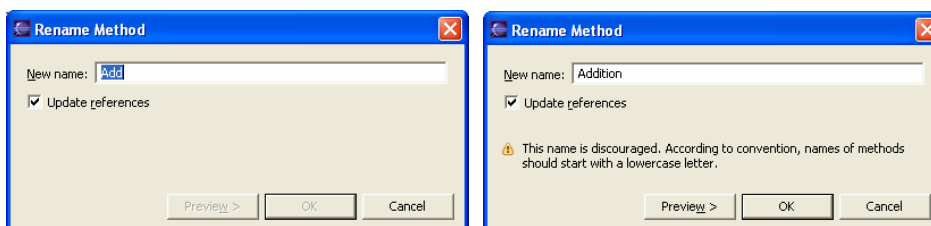
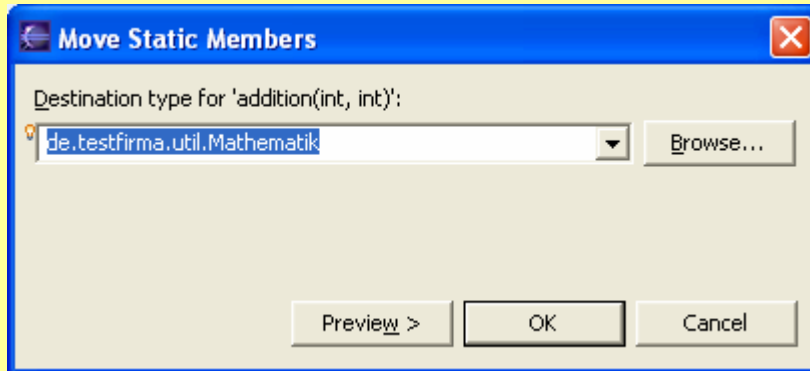


Abb. 6.10

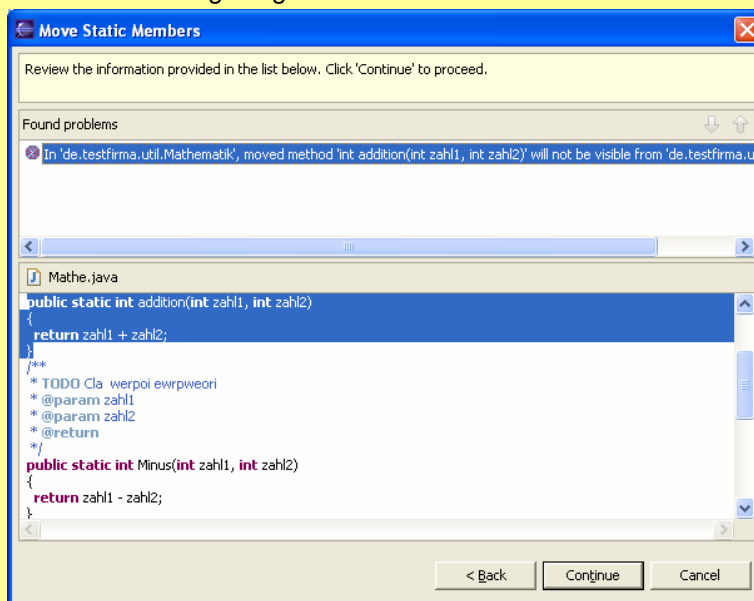
6.2 Methoden verschieben

Hiermit können Sie Methoden einer Klasse in eine andere verschieben.

- ✘ Markieren Sie die Methodendeklaration in rufen Sie den Menüpunkt REFACTOR - MOVE auf. Geben Sie den Typ (die Zielklasse) an oder wählen Sie den Typ über BROWSE aus.



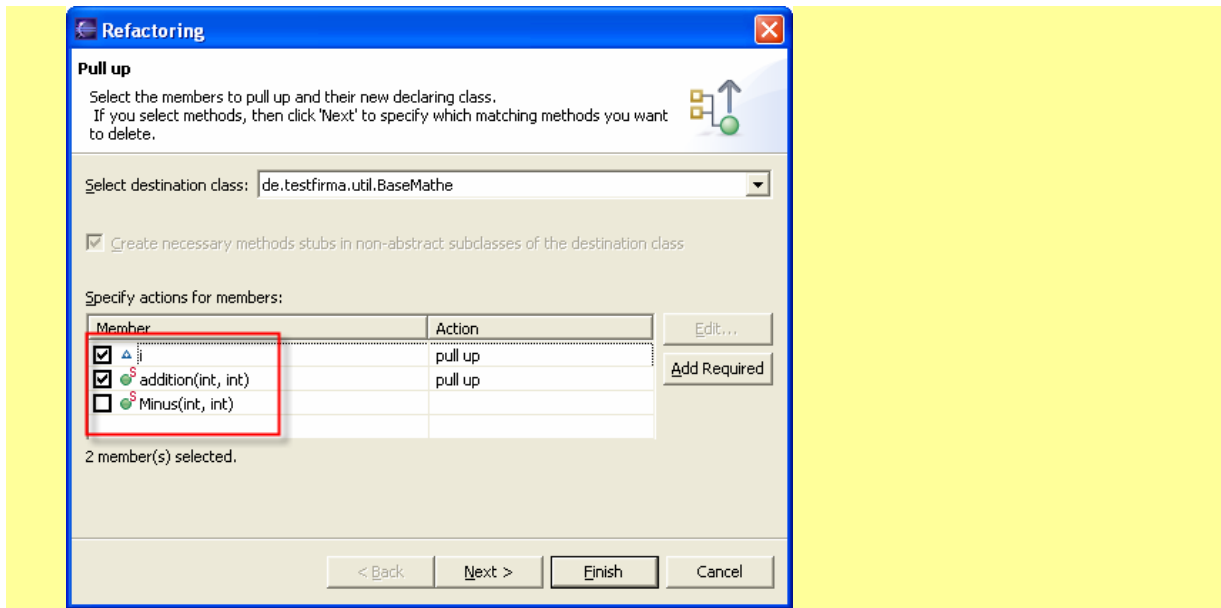
- ✘ Gibt es beim Verschieben Probleme mit existieren Code (im Beispiel ist die neue Klasse nicht für andere Klassen welche die Methode bereits verwenden sichtbar), wird ein Dialog mit mehr Informationen angezeigt.



6.3 Pull up - Member in Basisklassen verschieben

Mit diesem Refactoring verschieben Sie ausgewählte Member einer Klasse in eine Basisklasse.

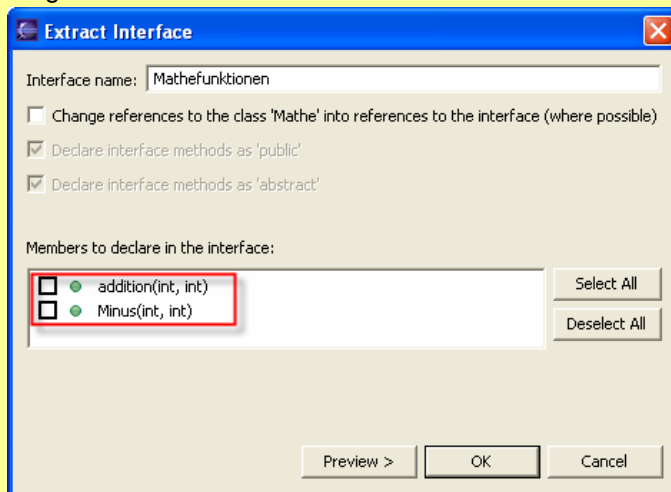
- ✘ Sie können sich dazu an einer beliebigen Stelle innerhalb der Klasse befinden und rufen den Menüpunkt REFACTOF - PULL UP auf.
- ✘ Markieren Sie im angezeigten Fenster die zu verschiebenden Member.



6.4 Schnittstellen extrahieren

Aus einer Klasse lassen sich für ausgewählte Methoden Interfaces extrahieren. Die Methoden müssen dazu `public` und nicht `static` sein.

- ✘ Rufen Sie innerhalb der Klasse den Menüpunkt REFACTOR - EXTRACT INTERFACE auf.
- ✘ Markieren Sie im Dialog die Methoden, die in das Interface aufgenommen werden sollen und vergeben Sie einen Namen für das Interface.



- ✘ Standardmäßig wird das Interface im selben Package wie die Klasse erzeugt. Außerdem wird die Klassendeklaration um `implements` erweitert.

Index

Anwendungen ausführen 11
Bezeichner umbenennen 21
Bibliothek erstellen 13
Download 3
Eclipse konfigurieren 5
Erster Start 3
Installation 3
Klasse erstellen 6
Konfiguration erzeugen 11
Methoden verschieben 22
Package Explorer 8
Perspektiven 5
Projekte 5
Projekte erstellen 5
Pull up 22
Refactoring 21
Schnittstellen extrahieren 23
Willkommenseite 4
Workbench 5
Workbench, Aufbau 7
Workspaces 5